2006 IEEE International Conference on
Systems, Man, and Cybernetics
October 8-11, 2006, Taipei, Taiwan

A Collaborating Team of Spiking Neural Network Based Robotic Agents for Inaccessible Fluidic Environments

Hani Hagras, *Senior Member, IEEE* , Martin Colley, Anthony Pounds-Cornish, Gustavo De Souza, Victor Callaghan, George Nikiforidis, Christos Argyropoulos, Achilles Kameas, Frank Murphy

***Abstract*— In this paper, we will introduce a novel system where identical miniaturized robotic agents with limited capabilities will collaborate to form a team that is capable of localizing and repairing scale formations in tanks and pipes within inaccessible fluidic environments. Each robotic agent is an autonomous entity that is based on the biologically inspired Spiking Neural Networks (SNNs) that communicate using pulses or spikes. The weights of the SNN are evolved using adaptive Genetic Algorithm (GA) that uses adaptive crossover and mutation to converge relatively fast to solutions that allow the robots to complete the desired tasks. The robotic agents communicate using indirect communication to move towards the site of scale formation and collaborate to repair damages.**

I. Introduction

Scale formation is a major problem in tanks and pipes within quiescent inaccessible fluidic environments [1]. The mineral components of scale deposits depend strongly on the chemical composition of the water used but in most cases it consists of calcium carbonate ($CaCO_3$) [1]. The formation of this mineral is favored by the high content of natural water in calcium salts and dissolved carbonate species and this formation can be described as follows [1]:

$$Ca^{2+}(aq) + HCO_3^-(aq) \rightarrow CaCO_3(s) + H^+ \quad (1)$$

The protons released from this reaction result in changes of the physicochemical parameters of the aqueous phase in contact with the steel walls of tanks or pipes and this promotes corrosion. Hence, the scale formation process is coupled with corrosion which will consequently cause severe damages [2].

Currently, the practice in industry is to add to the fluid chemical compounds that will effectively inhibit scale formation [2]. As a result of the fact that the action of the scale formation inhibitors is not localized, larger quantities of chemicals are needed and consequently the treatment cost increases [2]. Moreover, since in most of the cases these compounds are toxic, their disposal after use is also a problem [2]. In addition, as mentioned above, calcium carbonate is the most common component of scale deposit and once its quantity around the initial fault exceeds a critical threshold, an avalanche effect takes place causing its formation to become much quicker which will result in great difficulties in scale removal that will call for the use of highly-toxic substances [1], [2]. Therefore, early detection (before the avalanche takes place) and remediation of scale formation is imperative.

In this paper, we will introduce a novel solution based on using a team of miniaturised robotic agents for the localization and repair of scale formation. In the proposed solution, the robotic agents will be able to detect very early scale formation. The agents will then cooperate to move towards the fault and upon approaching the fault point they may infuse an effective scale formation inhibitor which shall result in stopping the scale development process right at the initial stages. The obvious advantage of this approach is that the scale inhabitation action shall be localized and hence more efficient. This will result in the use of very low quantities of the inhibitors which consequently provides a reduction in the operational costs and minimization of the environmental hazards.

Recently many miniaturized robots have been developed for navigation within inaccessible fluidic environments in industrial and medical applications [3], [4]. In our application, the robots need to have full autonomy which requires each of them to be capable of perceiving the environment, communicating effectively with the other robotic members of the team and being intelligent enough to make proper decisions in such a way the overall mission is achieved. However, these miniaturized robots will have limited memory, computation and power and hence the intelligence and communication modules used by these robots need to be able to operate on these limited platforms.

Biological neurons communicate by sending pulses across connections to other neurons [5]. The pulse is also known as a "spike" to indicate its short and transient nature [5]. Such neurons are called spiking neurons and their networks are termed Spiking Neural Networks (SNNs). As biological organisms have shown to be excellent control systems using SNNs then SNNs have the potential to produce good control systems for autonomous robots [6]. SNNs are deemed computationally more powerful than conventional artificial neural network formalisms on the basis of extensive theoretical work by Maass [7]. "Computationally more

This work was funded by the European Union (EU) Future and Emerging Technology programme under Grant IST-2001-38911 entitled "Self-Organised Societies of Connectionist Intelligent Agents capable of Learning"

H. Hagras, M. Colley, A. Pounds-Cornish, G. De Souza, V. Callaghan are with the Department of Computer Science, University of Essex, Wivenhoe Park, Colchester, CO4 3SQ, UK (phone: +44 1206 873601; fax: +44 1206 872788; e-mail: hani@essex.ac.uk).

G. Nikiforidis, C. Argyropoulos are with the Department of Medical Physics, School of Medicine, University of Patras, GR 26500, Rion, Greece

A. Kameas is with the Computer Technology Institute, Patras, Greece.

F. Murphy is with Tyndall National Institute, Cork, Ireland.

powerful" implies that SNNs need fewer nodes to solve the same problem than conventional artificial neural networks [7]. In addition, SNNs provide a number of other desirable features such as noise-robustness and simple real-world interfaces [8]. The computational power of SNNs exist because of the intrinsic time-dependent dynamics of spiking neurons that allow the temporal patterns of sensory-motor events to be captured and exploited more efficiently than the other connectionist models (i.e. with fewer neurons and simpler circuits) [5]. Moreover, SNNs can be mapped easily to hardware because the spikes are essentially binary events and the non linear dynamics and the coding of spiking circuits can be provided by spiking times, rather than by non linear, real valued activation functions [8]. In other words, a few logic operations and instructions to move around single bits over time would be sufficient to embed large circuits of spiking neurons that display complex abilities and behaviors into tiny and low power chips [6], [8]. Therefore such SNNs are appropriate control mechanisms for our miniaturized autonomous robots as they can give a very good response dealing with noise using tiny chips that consume little power.

A disadvantage of SNNs is that due to the non-continuous output function employed by such neuron models, standard learning algorithms based on gradient descent methods do not apply [9]. Attempts to modify back propagation and other methods have yielded little success [9]. Furthermore, the learning algorithms developed for SNNs are often restricted to very simple and application specific architectures [6], [9].

Artificial evolution through Genetic Algorithms (GAs) is therefore an interesting method to discover SNNs that autonomously develop desired behaviors for robots without imposing constraints on their architecture and functioning modality. GAs have been used to evolve signs of the weights (leaving the values of weights constants to 1) of the SNNs for robots behaviors as in [6]. In the initial phases of our project, we have used an adaptive online GA to evolve the weight values and signs of the SNNs in a relatively short time interval using real mobile robots interacting with their environment. It was shown that the evolved SNNs had given a very good performance in noisy and uncertain environments while outperforming other techniques like fuzzy logic control [10]. However, according to the authors' knowledge no work has evolved SNNs for miniaturized robots to navigate in 3D inaccessible fluidic environments which will be the focus of our work.

The communication between the robotic team is restricted by the physical limitations posed by the application domain and the limited computational infrastructure of the robots. Therefore the robots will use a simple communication scheme which is based on indirect communication rather than direct message passing.

In Section II, we will introduce the robotic agents and we will present their sensors and actuators as well as the indirect communication system and we will also present the agent team operation. Section III will present the used SNNs and their operation. Section IV will present the used adaptive GA. In Section V, we will present the experiments and results while conclusions and future work will be presented in Section VI.

II. The Robotic Agents

A. The Robots Description

The robotic agent team is composed of autonomous and identical robots where the real robotic agent is based on the submersible robot shown in Fig. 1a. The robot has roughly a spherical shape and it has a propulsion system for movement using pumped water jets. The robot is intended for use in collective underwater tasks where each robot has proximity sensors as well as 6 pH sensors to sense the pH concentration gradient (the pH change is maximum near the fault and it decreases away from the fault) which is correlated with the initiation of scale formation. The robot has 6 actuators to control which are four nozzles for actuation on the horizontal plane and two buoyancy actuators for actuation up and down along the vertical axis. The four nozzles actuate in the horizontal plane by expelling water drawn through an impeller at the bottom of the unit with a rotating collar selecting the active nozzle. A syringe draws or expels water through the bottom of the unit to control buoyancy thus actuating the unit along the vertical axis. The use of these actuators enables the robots to move in the 3D space. Each robot is equipped with a wireless RF communication module that is used to broadcast simple messages and to generate a gradient based on signal strength.

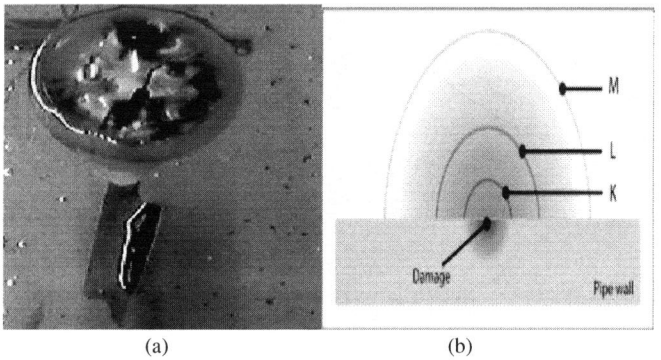
B. The Agents Indirect Communication

The communication among agents is an essential concept that gives rise to collaborative behaviors. In our system, the communication is restricted by the physical limitations imposed by the application domain and the limited computational abilities of the robots. Hence, the primary concern is to use a simple communication scheme that will reduce the technological challenges and give a simple robust protocol of limited information exchange that will result in agent collaboration. Due to these considerations, we have chosen the biologically inspired indirect communication that is based on the observed behavior of the other agents rather than direct message passing.

We empower the agents with indirect quorum sense communication by giving them a transmitter that emits a properly modulated wireless signal and a corresponding receiver that detects the signal emitted by other agents. Each agent can emit signals that can be perceived by other agents in the near vicinity depending on their relative location. However, there is no explicit or directed information communication. As although, the information is transferred by a radio signal for practical implementation reasons, it is not a deliberate act of communication between agents. The sender does not necessarily explicitly broadcast its state but allows others to observe it and the other agents may or may not be

able to receive the signal. The signal reception depends on the transmitter power and the receiver sensitivity. The transmitted signal's gradual attenuation will create the desired "gradient field" around the robot that emits the quorum signal.

The robots communication can be summarized as follows: in case there are no faults (no scale formation), the robotic agents will wander randomly and try to monitor the environment for any developing faults. When an agent perceives a fault signal, it will move towards its source (i.e. the fault area) and simultaneously releases a "quorum sense signal", which ultimately "attracts" other agents. When any of the other agents perceive the quorum signal, they move towards its source (i.e. the agent having perceived the fault signal) and also start emitting the quorum signal themselves. As quorum signals accrue, more agents are eventually attracted towards the fault area; when a quorum signal threshold is exceeded a spatio-temporally ordered community is formed in the vicinity of the fault and starts repairing it. The quorum signal threshold is proportional to the fault area as the larger the fault, the larger the size of the agent community used to repair this fault. When agents repair the fault the strength of the original environmental stimulus drops, quorum signals diminish and hence the community disperses.



(a) (b)

Fig. 1. (a) The robotic agent. (b) Regions K, L and M

C. The Operation of the Robotic Agents Team

The agents can be conceptually divided into three distinct teams corresponding to the spatial location at a specific moment and with regard to the proximity to the fault as shown in Fig. 1b.

The robotic agents are considered located in the inner region known as *region K*, if the robotic agents are located in an area close to the fault such that the fault signal sensed by the pH sensors exceeds a certain threshold. In this region the agent will try to move towards the gradient of the sensed "fault signal". The agent's state is transmitted via the wireless RF link along with quorum sense signal (a "fault_sensed" signal) thus indicating that the agent discovered a region of interest and it is now following a pH gradient. The gradient following strategy in this region is inspired from amoeboid cells [11]. The robotic agent controller for the pH gradient following in this region is based on SNNs that are evolved by the adaptive GA which will be described in the next sections.

The agents are considered located in the medium region known as *region L*, if the agents' pH sensors are outside the range of the "fault signal" but within the range of the RF signal transmitted by an agent within the *K region*. The RF antenna receives the "fault_sensed" RF signal and the agent starts moving towards the gradient of this signal. Inside the *L region*, the agent starts transmitting a "fault_reported" RF signal as soon as it receives the "fault_sensed" signal.

The agents are considered located in the outer region known as *region M* if they cannot receive the "fault_reported" RF signals transmitted by agents in *region L*. The *M region* agents wander randomly until they perceive some interesting signal to follow by going to *region L*.

III. Spiking Neural Networks

Networks of spiking neurons are very close to the real world biological neural network and they are capable of exploiting time as a resource for coding and computation in a much more sophisticated manner than virtually all other common computational models and this is responsible for the SNN computational power [5], [7].

The state of a spiking neuron is described by the voltage difference across its membrane, also known as membrane potential v [6]. Incoming spikes can increase or decrease the membrane potential. The neuron emits a spike when the total amount of excitation induced by the incoming excitatory and inhibitory spikes exceeds its firing threshold θ. After firing, the membrane potential of the neuron resets its state to a low negative voltage during which it cannot emit a new spike, and it gradually returns to its resting potential. The recharging period is called the *refractory period.*

There are several models of spiking neurons that account for these properties with various degrees of detail. In this paper we will use the *Spike Response Model* (SRM) [7]. In the SRM, the effect ε of an incoming spike on the neuron membrane is a function of the difference

$$s = t - t^f \tag{2}$$

Where t is the current time and t^f is the time when the spike was emitted (firing time). The properties of the function ε are determined by the following:

- The delay Δ between the generation of a spike at the pre-synaptic neuron and the time of arrival at the synapse.
- A synaptic time constant τ_s.
- A membrane time constant τ_m.

The idea is that a spike emitted by a pre-synaptic neuron takes some time to travel along the axon and once it has reached the synapse, its contribution to the membrane potential is highest as soon as it arrives but gradually fades as time passes [6]. A possible function $\varepsilon(s)$ describing this behavior is shown in Fig. 2a and can be written as follows [6]:

$$\varepsilon(s) = \exp\left[-(s-\Delta)/\tau_m\right]\left[1-\exp\left(-(s-\Delta)/\tau_s\right)\right] : s \geq \Delta$$
$$0 : s < \Delta \tag{3}$$

Once a neuron has emitted a spike, its membrane potential

is set to a very low value to prevent an immediate second spike and then it gradually recovers to its resting potential. The speed of recovery depends on the membrane time constant τ_m. A possible function $\eta(s)$, for this *refractory period* is shown in Fig. 2b and can be written as follows [6], [7].

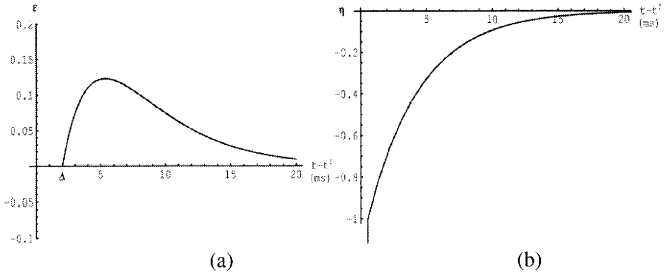
$$\eta(s) = -\exp[-s/\tau_m] \quad (4)$$

We can now put together the equations describing the synaptic contributions and the refractory period to describe the dynamics of a neuron that has several synaptic connections from the input neurons. Each synaptic connection has a weight w_{ij} which can be negative (inhibitory) or positive (excitatory). The membrane potential of a neuron i at time t_c is given by

$$v_i(t_c) = \sum_{j=1}^{N} w_{ij} \sum_{t=1}^{t_c} \varepsilon_j(s_j) + \sum_{t=1}^{t_c} \eta_i(s_i) \quad (5)$$

Where j is the pre-synaptic neuron and i is the post-synaptic neuron. w_{ij} is the weight of the synaptic connection between neuron i and neuron j. N is the total number of the pre-synaptic neurons. t_c is the current time. s_j is an application of (2) for the pre-synaptic neuron j and s_i is an application of (2) for the post-synaptic neuron i.

As it is complex to solve (5) [12], in each control cycle which takes T time steps we will iterate over t_c to find when (5) exceeds the threshold at the time the spike was emitted. In our robots the control cycle is 100 ms (according to the processing time of the used hardware) and each time step is 1 ms.



(a) (b)

Fig.2. (a) Function describing ε (b) Function describing refractory period.

In SNNs, a single spike is a binary event that can encode only the presence or absence of a stimulus. There are many ways of mapping the sensor's analog value to spikes at the beginning of the control cycle. One method consists of encoding the strength of the sensor value in the firing delay of the neuron. In this paper, we use this method for mapping the sensor's analog values to spikes. This coding system known as "delay coding" has been used by many researchers as it is simple and it is one of very few coding methods that might theoretically be used for very fast neural computation [12] which is required in our problem space.

For an analog input sensor value x_j to pre-synaptic neuron j, the firing time t_j^f can be calculated as follows [12]:

$$t_j^f = T - kx_j \quad (6)$$

Where T is the time of the control cycle and k is a suitable scaling factor. At the end of the control cycle, we need to convert the firing of the post-synaptic neuron i to analog outputs for the actuators. We are going to use the delay coding again, so the analog output y_i to the actuator connected to neuron i can be written as follows:

$$y_i = \frac{T - t_i^f}{c} \quad (7)$$

Where c is a suitable scaling factor, t_i^f is the firing time for the post-synaptic neuron i.

The robotic agents will use the SNN controllers to follow the pH gradient in the *K region*. The pH gradient following behavior will have inputs from 6 pH sensors and will control 6 actuators corresponding to 4 collar positions (for navigation in the horizontal plane) and two buoyancy controls (for navigation in the vertical plane). The generic architecture of a SRM SNN using delay coding is shown in Fig. 3, where in our SNNs, we use a two layer SNN, in which the pH sensors' analog value will be the inputs to the SNN's pre-synaptic neurons and the analog values from the post-synaptic neurons will be the outputs to the 6 actuators.

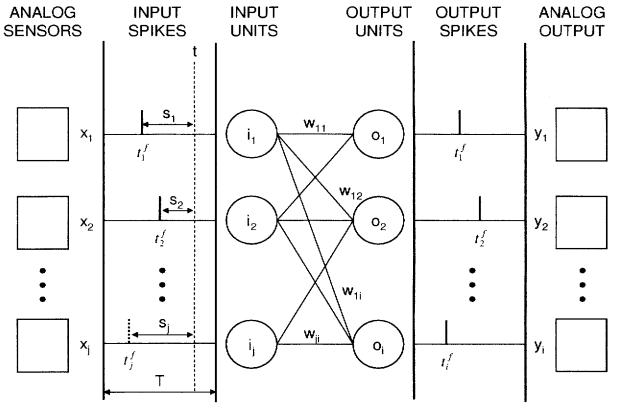


Fig.3. SRM SNN using delay coding architecture.

IV. Adaptive Genetic Algorithm

The GA will be used to evolve the values and the signs of the weights of the SNNs as well as the weight existence; the weights take any value between -1 and +1 where weights with zero value will be removed from the SNN to optimize its architecture. As the SNN is having 6 pH sensors and 6 actuators, therefore the fully connected SNN will have 36 weights to evolve. Therefore, the GA chromosome which represents a possible solution for the problem consists of all the 36 weights in the SNN. We used binary coding and we represented each weight by a small number of bits (3 bits) to reduce the chromosome length (chromosome length =3*36=108 bits). An n-bit string can accommodate all integers up to the value 2^n -1. So using 3 bits can represent 7 integer values, where a weight of +1 will be equivalent to an integer value of 7, a weight of -1 will be equivalent to an integer value of 0. In this way we can evolve the signs and values of the weights of the SNN. When testing a chromosome, the weight binary value is mapped back into a real weight value between -1 and +1 and applied to the SNN

controller which the robot uses to move in the *region K*. For the pH gradient following behavior, the fitness of each chromosome was evaluated according to how well the robot followed the pH gradient using the shortest path to arrive to the fault.

Standard GAs that use fixed crossover and mutation rates are widely known to be slow as they usually require big populations and they converge after a large number of generations. In this paper, we will use an adaptive GA which tries to maintain a limited amount of exploration and diversity in the population. These requirements mean that the chromosome and population size (we used a small population size of 50) should be kept relatively small, so that progression towards near-convergence can be achieved within a relatively short time [13]. Similarly the genetic operators (crossover and mutation) should be used in a way that rapidly achieves high-fitness individuals in the population [13]. In our adaptive GA, we will adaptively change the crossover and mutation probabilities based on Srinivas method [14]. This method helps us to achieve good crossover and mutation parameters that aid convergence in a short time interval. In order to vary P_c (crossover probability) and P_m (mutation probability) adaptively to prevent premature convergence of the GA, it is essential to be able to identify whether the GA is converging to an optimum [14]. One possible way of detecting convergence is to observe the average fitness value f' of the population in relation to the maximum fitness value f_{max} of the population. $f_{max} - f'$ is likely to be less for a population that has converged to an optimum solution than that for a population scattered in the solution space. P_c and P_m are defined as follows [14]:

$$P_c = \frac{f_{\max} - f''}{f_{\max} - f'} : f'' \geq f'$$

$$P_c = 1 : f'' < f' \tag{8}$$

$$P_m = \frac{f_{\max} - f}{2(f_{\max} - f')} : f \geq f'$$

$$P_m = 0.5 : f < f' \tag{9}$$

Where f'' is the larger of the fitness values of the solutions to be crossed. f is the fitness of the individual solutions. The method means that we have P_c and P_m for each chromosome. The type of crossover was chosen to be a one point crossover for computational simplicity and real time performance.

One of the goals of this approach is to prevent the GA from getting stuck in a local optimum. For the average and sub average fitness chromosomes, we employ a high P_m value of 0.5 to introduce new genetic material without reducing the search process to a random process [14]. The same for the P_c which takes a value of 1.0 to ensure that average and sub average fitness chromosomes undergo crossover.

V. Experiments and Results

In order to test and evaluate the various concepts of the project, we will use graded testing platforms which will start by a real world 3-D simulator to establish all the project concepts and evaluate their success. After this, we will try to bridge the gap between simulation and the real world by linking the 3-D simulation environments to the robots hardware, sensors and actuators to be part of the simulation environments to form a hardware in the loop simulation. This will help to develop realistic controllers and communication modules for the robotic agents so that they can operate successfully in the real test bed shown in Fig. 4a which consists of a Plexiglass™ tube of 2m long and 50cm diameter.

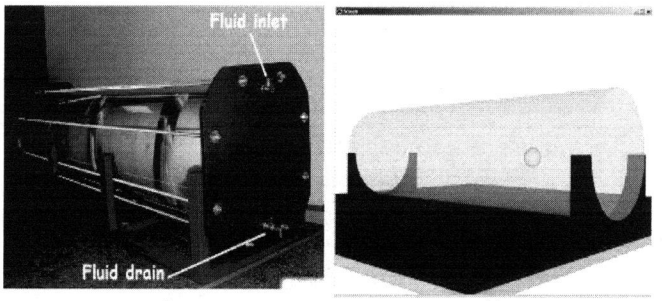


Fig.4. (a) The real test bed. (b) The 3D simulator of the test bed.

This section will report on the experiments and results obtained from the first test bed of the project within the simulation environment shown in Fig. 4b. The developed 3D simulation environment includes a realistic physics model, an accurate real-world scale and the ability to log the status of every agent in terms of position, sensor and actuator values time stamped to the nearest millisecond. The simulator instantiates a TCP socket server that allows robot controllers running in remote machines or written in different programming languages to interact with it. In this way, raw sensor data is accessed by controllers through a proxy and actuator commands are passed back to the simulation engine. We also developed a proxy to the simulation engine that acts as a serial interface wrapper (i.e. RS232) which enables us to exchange data between the simulator and the hardware implementations of the controllers (i.e. FPGA boards) making hardware in the loop simulation possible.

Due to the limited space we will only present proof of concept experiments to the main project components.

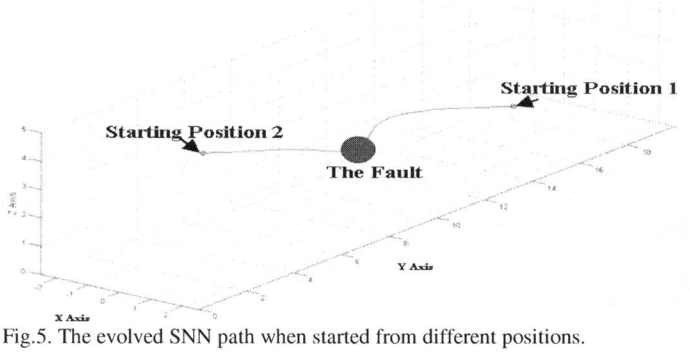
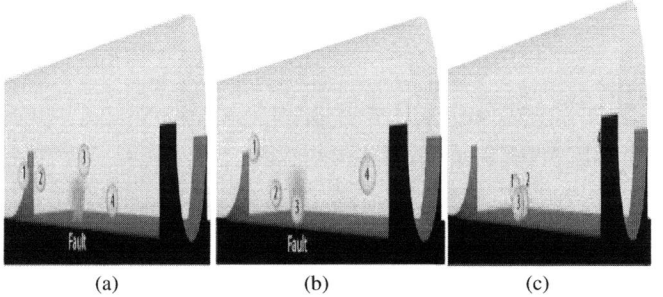


Fig.5. The evolved SNN path when started from different positions.

Fig. 5 shows one of the partially evolved SNN paths when started from different starting positions. For each starting position, we calculated the average and standard deviations

from the optimum path (which correspond the straight path between the starting position and the fault). The robot was always able to follow successfully the pH gradient to the fault with average deviation of 1.6 cm and a standard deviation of 0.7 from the optimum path.

In order to demonstrate the success of the agent communication and collaboration to repair faults, we show in Fig. 6 one representative scenario from the numerous scenarios we have implemented and tested. Fig.6 illustrates the robotic team's capability to cooperate in order to detect and repair a single fault in a pipe filled with quasi-static fluid. The size of the fault area in Fig.6 will need three robotic agents to repair (this will vary with different sizes of the fault area). The 4 robots shown in Fig.6 were wandering randomly in the environment monitoring for possible faults before the introduction of the fault associated with the scale formation which resulted in the creation of the pH gradient. Fig.6a shows a snapshot of the environment at a particular time t_1, where three robots are in the M region (robots 1,2,4) surveying the environment while robot 3 has detected a fault in *region K* and heads towards the fault where it transmitted via the wireless RF link a "fault_sensed" signal. Fig.6b depicts the situation at t_2, where robots 1 and 2 are in *region L* as robot 2 received the "fault_sensed" signal first and transmitted the "fault_reported" RF signal which is picked by robot 1 and hence both robots will follow their sensed RF gradient till they enter *region K*. Fig. 6c shows a snapshot of the environment at t_3 where robots 1,2,3 arrive to the fault and repair it and hence the pH gradient disappears and the robots continue wandering the environments. Note that in this experiment robot 4 was always in region M as it did not receive either the pH or RF signals.



(a) (b) (c)

Fig.6. Agents collaboration to repair faults.

VI. Conclusions

In this paper, we have introduced a novel framework for collaborating multiple robotic agents that operate within inaccessible environments. Each robotic agent has limited computation and battery power and consequently SNNs were chosen to provide efficient autonomous navigation to follow the pH gradient within the *K region* and arrive to the fault. The weights of the SNN were evolved using adaptive GA that can converge relatively fast (while escaping local minima) to suitable solutions that will allow the robots to complete their desired tasks. The members of the agent team communicate and collaborate using indirect communication to move towards the site of fault in order to repair damages. Although the application of our work is the localisation and repair of scale formations in tanks and pipes within inaccessible fluidic environments, the concepts of the project can be extended to operations within other inaccessible environments such as deep water operation, cleanup of nuclear waste, etc.

This paper has presented experiments and results evaluating the success of the main project concepts within a true 3-D fluidic simulation. We are currently working on testing the whole system within the real test bed shown in Fig. 4a and we will report on these results in subsequent publications.

Acknowledgment

We would like to thank John Hallam for his great help and support with the Hydron robots used in this project.

References

[1] J. Cowan and D. Weintritt, *Water-formed scale deposits*, Huston, Texas: Gulf Pub. Co, 1976.

[2] M. Fontana, *Corrosion Engineering*, New York: McGraw Hill, 1996.

[3] T. Fukuda, A. Kawamoto, F. Arai, and H. Matsuura, "Steering Mechanism of Underwater Micro Mobile Robot in Water," *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, Nagoya, Japan, May 1995, pp. 363-368.

[4] S. Guo, T. Fukuda, and K. Asaka, "A new type of fish-like underwater micro robot," *IEEE/ASME Transactions on Mechatronics*, vol. 8, pp.136-141, 2003.

[5] W. Maass, "Networks of spiking neurons: the third generation of neural network models," *Transactions of Interational Society for Computer Simulation*, vol. 14, pp. 1659-1671, 1997.

[6] D. Floreano and C. Mattiussi, "Evolution of spiking neural controllers for autonomous vision-based robots," *Proceedings of the International Symposium on Evolutionary Robotics*, 2001, pp. 38-61.

[7] W. Maass, "On the computational complexity of networks of spiking neurons," in Advances in Neural Information Processing Systems, vol. 7, 1995, pp. 183-190.

[8] T. Lehmann and R. Woodburn, "Biologically-inspired on-chip learning in pulsed neural networks," in Analog Integrated Circuits and Signal Processing, vol. 18. 1999, pp. 117-131

[9] H. Burgsteiner, "Training networks of biological realistic spiking neurons for real-time robot control," *Proceedings of the 9th International Conference on Engineering Applications of Neural Networks*, Lile, France, August 2005.

[10] H. Hagras, A. Pounds-Cornish, M.Colley, V.Callaghan and, G. Clarke, "Evolving Spiking Neural Network Controllers for Autonomous Robots, " *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, New Orleans, USA, April 2004, pp. 4620-4626.

[11] J. Dusenbery, "Performance of Basic Strategies for following Gradients in Two Dimensions," *Journal of theoretical Biology*, vol. 208, pp. 345-360, 2001.

[12] B. Tonkes, "Simulation issues in spiking neural networks," *Proceedings of the 8th Australian Conference on Neural Networks*, 1997, pp. 80-84.

[13] G. Linkens and O. Nyongeso, "Genetic algorithms for fuzzy control, part II: online system development and application," *IEE Proceedings on Control Theory Applications*, pp. 177-185, 1995.

[14] M. Srinivas and L. Patnaik, "Adaptation in Genetic Algorithms," in Genetic Algorithms for Pattern Recognition, S. Pal and P. Wang, Eds. Florida: CRC Press, 1996, pp.45-64.